Implicit Methods In CFD[1]

Thomas H. Pulliam
Research Scientist, CFD Branch
NASA Ames Research Center

November 6, 1996

[1]Presented at ICFD 1988 Conference on Numerical Methods for Fluid Dynamics, Oxford University, Oxford, UK, March 21-24, 1988

Abstract

Various forms of implicit finite difference schemes are examined from the point of view of Newton's method or modified forms of Newton's method for steady state convergence. The class of schemes examined include central plus artificial dissipation methods, upwind/TVD schemes, and splitting methods (e.g. flux vector, upper/lower and eigensystem decompositions). The paper mainly concentrates on analysis of the schemes and addresses issues such as vectorization, efficiency and accuracy. The effect of approximate factorization, artificial dissipation and operator splitting on stability will be discussed. The emphasis will be on the Euler equations.

1.1 Introduction

1.1.1 Discussion

Implicit finite difference techniques have taken the dominant role in the computation of fluid dynamics and aerodynamics. Over a decade ago one could have made the observation that a substantial portion of all computational results were obtained using explicit techniques such as Lax-Wendroff or most likely MacCormack's scheme [1], (the remainder came under the classification of analytic, spectral or perturbation schemes). At that time the limitations of computer speed and algorithm development made the explicit schemes the most practical. Explicit schemes are easy to program, requiring no additional numerical algorithms. In contrast an implicit scheme invariably leads to the need to solve either linear or nonlinear systems of equations. Linear algebra plays a major role in the solution process, where large sparse (but usually structured) matrices are inverted. Until recently, Gaussian elimination (or LU decompositions such as the Thomas algorithm) was commonly used to invert the matrices. With the advent of parallel and vector computer architectures, sparse matrix solvers may be the optimal way to proceed. Methods such as cyclic reduction, nested dissection, and sparse factorization techniques may take over especially if they can be more efficient on the new generation of high speed processors.

For a given time accuracy, explicit methods can require significantly less computational operations then implicit schemes and at least at first glance explicit methods seem to be easier to vectorize. Implicit methods have the advantage of being typically unconditionally stable (in terms of linear analysis) as compared with limited stability for explicit schemes. In cases where time accuracy restricts one to time steps on the order of or less than the stability bounds of an explicit method, such methods may be more efficient than a corresponding implicit scheme. In general though, even within the limit of time accuracy, explicit schemes can be more restrictive in terms of the allowable time step than is warranted by the time scales of the solution. For instance, boundary layer mesh scales may be more restrictive than the spatial scales of a plunging body. Implicit schemes have been used more extensively for steady state problems where the restriction of a time step of integration commensurate with the physics of a problem is ignored in the hope of converging the solution more rapidly. In fact, both explicit and implicit schemes commonly employ time steps which vary across the physical domain (local time steps) so that time accuracy is completely lost. In these cases, implicit schemes have an advantage over explicit schemes. The one exception where explicit based schemes have performed as well or better than implicit schemes for steady state is in the case of multigrid acceleration of multi-stage integration, see [1, 2]. Multigrid has been applied to both inviscid [1] and viscous cases [3] and has even been analyzed for unsteady applications [4]. It should be noted that even in those schemes some form of an implicit operator is used (residual averaging).

1.1.2 Equations

The methods by which schemes are made implicit can take a wide variety of forms in terms of efficiency, consistency, accuracy and resulting stability and convergence. In this paper we shall examine various forms of implicit algorithms which can be applied to the compressible Euler and Navier-Stokes equations. The starting point is a set of partial differential equations, (in this case the Euler equations in conservation law form)

$$\partial_t Q + \partial_x E + \partial_y F + \partial_z G = 0 \tag{1.1}$$

where

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{bmatrix}, \quad E = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(e+p) \end{bmatrix}, \quad F = \begin{bmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vw \\ v(e+p) \end{bmatrix}, \quad F = \begin{bmatrix} \rho w \\ \rho wu \\ \rho wv \\ \rho w^2 + p \\ w(e+p) \end{bmatrix} \tag{1.2}$$

Pressure is related to the conservative flow variables, Q, by the equation of state

$$p = (\gamma - 1)\left(e - \frac{1}{2}\rho(u^2 + v^2 + w^2)\right)$$

where γ is the ratio of specific heats, generally taken as 1.4. The speed of sound is a which for ideal fluids is given by, $a^2 = \gamma p/\rho$.

In the presentation below we will periodically be switching between the full nonlinear system given above and various reductions and simplifications. In some cases we will just consider the one-dimensional ($v = w = 0$) and two-dimensional forms ($w = 0$). We shall also employ a representative scalar form where we have

$$u_t + f(u) = 0$$

with either $f(u) = au_x + bu_y + cu_z$ the scalar wave equation, or $f(u)$ a general nonlinear function of u, which could be multi-dimensional, having similar definitions as needed.

If we examine the $1 - D$ wave equation

$$u_t + u_x = 0$$

and apply Fourier analysis $u(x,t) = w(t)e^{i\theta x}$, for spatial difference operators, such as central differencing we have

$$u_t + u_x = u_t + \delta_x u_j = u_t + \frac{u_{j+1} - u_{j-1}}{2\Delta x} \rightarrow w_t + \frac{i \sin\theta\Delta x}{\Delta x} w = w_t + \lambda_x^c w = 0$$

or for 1st order backward

$$\begin{aligned} u_t + u_x = \quad & u_t + \nabla_x u_j = u_t + \tfrac{u_j - u_{j-1}}{\Delta x} \rightarrow \\ w_t \quad & + \tfrac{1 - \cos\theta\Delta x + i\sin\theta\Delta x}{\Delta x} w = w_t + \lambda_x^b w = 0 \end{aligned}$$

When we talk of the differencing signature we will be referring to the eigenvalue λ_x associated with the choice of spatial differencing. This is generally termed the semi-discrete eigenvalue of the partial differential equation. In the above, λ_x can be either pure imaginary or complex, which indicates that we should consider complex $f(u)$ in our analysis since spatial differences will produce complex eigensystems.

There are a number of issues to be examined when developing an implicit algorithm for Eq. (1.1). First, a spatial differencing scheme must be chosen. Within the framework of finite difference schemes we shall consider central and upwind differencing of the flux terms, $\partial_x E$ and $\partial_y F$. In Section 1.2 , we shall motivate the choices, discuss various splittings of the flux vectors E and F which lead to flux vector and flux difference schemes and provide the framework for the implicit methods.

As the second component in developing an implicit scheme, we adopt a Newton like approach to the development, assessment and analysis of implicit operators. In Section 1.3 we discuss the equivalence of an implicit operator to an approximate Newton scheme. Various approximations are analyzed using an eigensystem analysis technique introduced in [5], which is a periodic Neumann-like approach where the system coupling is maintained leading to generalized eigensolutions.

1.2 Newton's Method

Newton's method can be used to understand the approximations used in implicit schemes. If we start out looking at Newton's method for the fixed point problem

$$F(Q) = 0 \tag{1.3}$$

we first do a Taylor series expansion with remainder of the function $F(Q)$ about some iteration level Q^n which leads to

$$F(Q) = F(Q^n) + \frac{\partial F}{\partial Q}^n (Q - Q^n) + \frac{\partial^2 F}{\partial Q^2}^* (Q - Q^n)^2 \tag{1.4}$$

where Q^* lies in the solution interval between Q and Q^n. Note that by $\frac{\partial F}{\partial Q}^n$ we mean the matrix Jacobian of $F(Q)$ with respect to Q evaluated at Q^n.

Dropping the remainder term (leading to a first order approximation) and evaluating Eq. (1.4) at $n+1$, assuming $F(Q^{n+1}) \approx 0$ gives the iterative scheme

$$Q^{n+1} = Q^n - \left[\frac{\partial F^n}{\partial Q}\right]^{-1} F(Q^n)$$

which is Newton method for a nonlinear system.

It is easy to show that Newton's method is quadratically convergent for any initial solution in the domain of attraction. Using the scalar form of Eq. (1.3), Eq. (1.4)

$$f(u) = 0 : \quad u^{n+1} = u^n - \frac{f(u^n)}{f'(u^n)} = g(u^n) \tag{1.5}$$

let

$$f(v) = 0 : \quad v \quad \text{the root}$$

The iterative scheme Eq. (1.5) is convergent if $|g'(u)| < 1$. We have

$$g'(u) = \frac{f''(u)f(u)}{[f'(u)]^2}$$

Expanding $g(u^n)$ in a Taylor series as above, with ξ between u^n and v, we have

$$g(u^n) = g(v) + g'(v)(u^n - v) + \frac{g''(\xi)}{2}(u^n - v)^2 \tag{1.6}$$

Letting the error be defined as $e^n = u^n - v$ and using $g(v) = v$, and $g'(v) = 0$ we have

$$e^{n+1} = u^{n+1} - v = g(u^n) - v = \frac{g''(\xi)}{2}(e^n)^2$$

which shows quadratic convergence.

One of the most commonly used implicit schemes is the 1^{st} order Euler implicit method. Writting Eq. (1.1) for now as

$$Q_t + F(Q) = 0$$

we have

$$\frac{(Q^{n+1} - Q^n)}{h} = -F(Q^{n+1}) \tag{1.7}$$

with $h = \Delta t$.

The function $F(Q)$ is nonlinear in Q and therefore we linearize Eq. (1.7) about time level $n+1$ which results in

$$\left[\frac{1}{h}I + \frac{\partial F^n}{\partial Q}\right]\Delta Q^n = -F(Q^n)$$

with $\Delta Q^n = (Q^{n+1} - Q^n)$, the so called "delta" form of the implicit scheme. Rearranging terms we get

$$Q^{n+1} = Q^n - \left[\frac{1}{h}I + \frac{\partial F^n}{\partial Q}\right]^{-1} F(Q^n)$$

which in the limit as $h \to \infty$ reduces to Eq. (1.4), Newton's method. For a finite h we have an approximate Newton scheme.

Analyzing the scalar form of the Euler implicit scheme as we did for full Newton, we have

$$u_t + f(u) = 0 : \quad u^{n+1} = u^n - \frac{hf(u^n)}{1 + hf'(u^n)} = g(u^n)$$

again letting

$$f(v) = 0 : \quad v \quad \text{the root}$$

Continuing as before

$$g'(u) = 1 - \frac{hf'(u)}{1 + hf'(u)} + h^2 \frac{f''(u)f(u)}{[1 + hf'(u)]^2} \tag{1.8}$$

Using Eq. (1.6), $f(v) = 0, g(v) = v$ and Eq. (1.8) evaluated at v we have

$$g'(v) = \frac{1}{1 + hf'(v)} = \sigma(h)$$

which gives

$$e^{n+1} = \sigma(h)e^n + \frac{g''(\xi)}{2}(e^n)^2$$

We will consider $f'(u)$ to be either real or complex (it could be complex if $f(u)$ is derived from a differential operator). Note then that for small $h : \sigma(h) \approx 1$ and convergence is linear, but as $h \to \infty : \sigma(h) \to 0$ and we approach quadratic convergence.

As the above example demonstrates, approximations to Newton's method can lead to slower convergence, and in fact sometimes instability (although we shall restrict ourselves to examining schemes which are stable). In general, an implicit scheme has some equivalence to a modified Newton's method and we shall be analyzing schemes from that point of view.

As we have seen the Euler implicit scheme is a good approximation to Newton's method producing good convergence characteristics for large time steps. Other approximations usually inhibit the convergence and we shall be examining various forms below. There is also a class of problems where we are interested in time accurate computations. In those cases, though, we usually start with an initial solution which generates a transient which is hopefully eliminated at some time and then an accurate resolution of the time variations is obtained. One candidate would be the 2nd order accurate trapezoidal scheme, here written for the scalar equation $u_t + f(u) = 0$ as

$$\frac{u^{n+1} - u^n}{h} + \frac{f(u^{n+1}) + f(u^n)}{2} = 0$$

which after expanding $f(u^{n+1}) = f(u^n) + f'(u^n)(u^{n+1} - u^n) + O(h^2)$ leads to the 2^{nd} order scheme

$$u^{n+1} = u^n - \frac{2hf(u^n)}{2 + hf'(u^n)} = g(u^n)$$

For this we have

$$g'(u) = 1 - \frac{2hf'(u)}{2 + hf'(u)} + \frac{2hf''(u)f(u)}{[2 + hf'(u)]^2}$$

where using Eq. (1.6), $f(v) = 0$ and $g(v) = v$ gives

$$e^{n+1} = -e^n + \frac{g''(\xi)}{2}(e^n)^2$$

which is nonconvergence, i.e. there is no reduction in the error with iteration. In general the trapezoidal scheme is unsuitable for steady-state and usually not desirable for time accurate problems because of this lack of error reduction.

1.3 Factorizations, Splitting And Approximations

Applying the Euler implicit time differencing to the two-dimensional form of the Euler equations (1.1) we have

$$\frac{(Q^{n+1} - Q^n)}{h} + E_x^{n+1} + F_y^{n+1} = 0$$

Linearizing the fluxes gives

$$\begin{aligned} E^{n+1} = & \; E^n + A^n (Q^{n+1} - Q^n) + O\left((Q^{n+1} - Q^n)^2\right) \\ F^{n+1} = & \; F^n + B^n (Q^{n+1} - Q^n) + O\left((Q^{n+1} - Q^n)^2\right) \end{aligned}$$

where the Jacobian matrices $A = \frac{\partial E^n}{\partial Q}$ or $B = \frac{\partial F^n}{\partial Q}$ are given by

$$\begin{bmatrix} 0 & \kappa_x & \kappa_y & 0 \\ -u\theta + \kappa_x\phi^2 & \theta - (\gamma - 2)\kappa_x u & \kappa_y u - (\gamma - 1)\kappa_x v & (\gamma - 1)\kappa_x \\ -v\theta + \kappa_y\phi^2 & \kappa_x v - (\gamma - 1)\kappa_y u & \theta - (\gamma - 2)\kappa_y v & (\gamma - 1)\kappa_y \\ \theta[\phi^2 - a_1] & \kappa_x a_1 - (\gamma - 1)u\theta & \kappa_y a_1 - (\gamma - 1)v\theta & \gamma\theta \end{bmatrix}$$

with $a_1 = \gamma(e/\rho) - \phi^2$, $\theta = \kappa_x u + \kappa_y v$, $\phi^2 = \frac{1}{2}(\gamma - 1)(u^2 + v^2)$, and $\kappa_x = 1$, $\kappa_y = 0$ for A or $\kappa_x = 0$, $\kappa_y = 1$ for B. These are 3×3 matrices in one-dimension (1-D), 4×4 in 2-D, and 5×5 in 3-D. Their exact form for the Euler equations can be found in numerous papers, e.g. [6].

Using the "delta" form leads to

$$[I + h\delta_x A^n + h\delta_y B^n]\left(Q^{n+1} - Q^n\right) = -h\left(\delta_x E^n + \delta_y F^n\right) \qquad (1.9)$$

where δ represents a derivative operator which can be either analytic or numerical.

Assuming the use of 2nd order central differences for δ, the solution of Eq. (1.9) requires the inversion of a large sparse banded matrix.

1.3.1 Matrix Form of Unfactored Algorithm

If central differences are used in Eq. (1.9) it is easy to show that the implicit algorithm produces a large banded system of algebraic equations. Let written as the mesh size in x be $Jmax$ and in y by $Kmax$. We choose an ordering of the data with the j index running first and the k index second, other orderings are just permutations of the data. Then the banded matrix is a $(Jmax \cdot Kmax \cdot 4) \times (Jmax \cdot Kmax \cdot 4)$ rectangular matrix of the form

$$\begin{bmatrix}
[] & [] & & & [] & & & & & & & \\
[] & [] & [] & & & [] & & & & & & \\
 & [] & [] & [] & & & [] & & & & & \\
 & & [] & [] & & & & [] & & & & \\
[] & & & & [] & [] & & & [] & & & \\
 & \ddots & & & \ddots & \ddots & \ddots & & & \ddots & & \\
 & & -B & & & -A & I & A & & & B & \\
 & & & \ddots & & & \ddots & \ddots & \ddots & & & \ddots \\
 & & & [] & & & [] & [] & & & & [] \\
 & & & & [] & & & & [] & [] & & \\
 & & & & & [] & & & [] & [] & [] & \\
 & & & & & & [] & & & [] & [] & [] \\
 & & & & & & & [] & & & [] & []
\end{bmatrix}$$

The matrix is sparse but it would be very expensive (computationally) to solve the algebraic system. For instance, for a reasonable two-dimensional calculation of transonic flow past an airfoil we could use approximately 80 points in the x direction and 40 points in the y direction. The resulting algebraic system is a 12,800 $\times$ 12,800 matrix problem to be solved and although we could take advantage of its banded sparse structure it would still be very costly in both CPU time and storage.

1.3.2 Approximate Factorization

As we have seen, the integration of the full two-dimensional operator is too expensive. One way to simplify the solution process is to introduce an approximate factorization of the two-dimensional operator into two one-dimensional operators. The implicit side (left hand) of Eq. (1.9) can be written as

$$\begin{aligned}[I + h\delta_x A^n + h\delta_y B^n] \quad & \Delta Q^n = \\ [I + h\delta_x A^n] \quad & [I + h\delta_y B^n]\,\Delta Q^n - h^2 \delta_x A^n \delta_y B^n \Delta Q^n \end{aligned} \tag{1.10}$$

The cross term (h^2 term) is second order in time since ΔQ^n is $O(h)$. It can therefore be neglected without degrading the time accuracy of any second order scheme which we may choose.

The resulting factored form of the algorithm is

$$[I + h\delta_x A^n]\,[I + h\delta_y B^n]\,\Delta Q^n = -h\,[\delta_x E^n + \delta_y F^n] \tag{1.11}$$

We now have two implicit operators each of which is block tridiagonal. The structure of the block tridiagonal matrix is

$$\begin{bmatrix} [\,] & [\,] & & & & & & \\ [\,] & [\,] & [\,] & & & & & \\ & [\,] & [\,] & [\,] & & & & \\ & & \ddots & \ddots & \ddots & & & \\ & & & -A & I & A & & \\ & & & & [\,] & [\,] & [\,] & \\ & & & & & [\,] & [\,] & [\,] \\ & & & & & & [\,] & [\,] \end{bmatrix}$$

The solution algorithm now consists of two one-dimensional sweeps, one in the x and one in the y direction. The block matrix size is now at most $(\max[Jmax, Kmax] \cdot 4) \times (\max[Jmax, Kmax] \cdot 4)$. Each step requires the solution of a linear system involving a block tridiagonal which is solved by block LUD (lower-upper decomposition). The resulting solution process is much more economical than the unfactored algorithm in terms of computer storage and CPU time.

1.3.3 Newton Scalar Analysis of Factored Implicit Scheme

The use of factorization produces an efficient and practical algorithm, but as we shall see the use of the factorization leads to a reduction in the convergence properties and sometimes stability of the resulting algorithm. First we shall use the scalar Newton analysis as above and then introduce a model system analysis which retains the structure of the Euler equations. A representative scalar model of the factored algorithm is

$$u_t + f_1(u) + f_2(u) = u_t + r(u) = 0$$

which leads to

$$[1 + hf_1'(u^n)]\,[1 + hf_2'(u^n)]\left(u^{n+1} - u^n\right) = -hr(u^n) \qquad (1.12)$$

where we are seeking solutions to $r(v) = 0$. Note that the "delta form" guarantees that the steady solution $r(v) = 0$ is satisfied independent of h (Δt). We recast Eq. (1.12) as

$$u^{n+1} = u^n - \frac{hr(u^n)}{[1 + hf_1'(u^n)][1 + hf_2'(u^n)]} = g(u^n)$$

$$g'(u) = 1 - \frac{hr'(u)}{[1+hf_1'(u)][1+hf_2'(u)]} + \frac{h^2 r(u) f_1''(u)}{[1+hf_1'(u)]^2[1+hf_2'(u)]} + \frac{h^2 r(u) f_2''(u)}{[1+hf_1'(u)][1+hf_2'(u)]^2}$$

Using the definitions and Eq. (1.6) of Section 1.2, we have

$$e^{n+1} = \sigma(h)e^n + \frac{g''(\xi)}{2}(e^n)^2$$

with

$$\sigma(h) = \frac{1 + h^2 f_1'(v) f_2'(v)}{[1 + hf_1'(v)][1 + hf_2'(v)]}$$

Now for real or complex $f_1'(v)$ and $f_2'(v)$ the coefficient $\sigma(h)$ of e^n is bounded by 1 for all h and as $h \to \infty$, $\sigma(h) \to 1$ showing that for large h we approach nonconvergence. For small h it can be shown that $\sigma(h) < 1$ and the scheme does converge and in fact there is an optimal h for maximum convergence.

We can next look at the equivalent of a three-dimensional factorization using

$$u_t + f_1(u) + f_2(u) + f_3(u) = u_t + r(u) = 0$$

Forming the iterative scheme

$$u^{n+1} = u^n - \frac{hr(u^n)}{[1 + hf_1'(u^n)][1 + hf_2'(u^n)][1 + hf_3'(u^n)]} = g(u^n)$$

which gives

$$e^{n+1} = \sigma(h)e^n + \frac{g''(\xi)}{2}(e^n)^2$$

with

$$\sigma(h) = \frac{1+h^2 f_1'(v)f_2'(v)+h^2 f_2'(v)f_3'(v)+h^2 f_1'(v)f_3'(v)+h^3 f_1'(v)f_2'(v)f_3'(v)}{[1+hf_1'(v)][1+hf_2'(v)][1+hf_3'(v)]}$$

Now again in this case we have $\sigma(h) \to 1$ as $h \to \infty$, but in the case of complex f_i we also have that for some $h > h_c$: $\sigma(h) > 1$ which implies that the three-dimensional factored algorithm is at best conditionally stable. In fact, consider the three-dimensional wave equation solved using second order central differences. In that case, the differencing signature ($\lambda's$) produces pure imaginary f_i and the scheme would then be unconditionally unstable. Any real part occurring in the f_i would be associated with dissipative differencing or the addition of a dissipative term, see discussion on artificial dissipation below.

1.3.4 Newton System Analysis of Factored Implicit Scheme

The scalar analysis presented above provides guidelines for algorithm choices and gives useful information about the characteristics of the resulting schemes. We can go one step further in understanding the application to systems, such as the Euler equations, by employing an analysis technique which maintains the system structure, see [5]. As an example of the usefulness of such an analysis, consider Eq. (1.9). The left hand side of that equation could be reduced if we could simultaneously diagonalize A and B, thereby reducing the system to 4 scalar operators instead of a 4×4 block operator. This can not be done since A and B do not commute (they don't have a common set of eigenvectors). For the unfactored algorithm this complicated the computational work, but in the case of the factored algorithm it also has an impact on the stability and accuracy since the order of factorization, see Eq. (1.10), can produce a cross term of either $h^2\delta_x A^n \delta_y B^n (Q^{n+1} - Q^n)$ or $h^2\delta_y B^n \delta_x A^n (Q^{n+1} - Q^n)$ which can be significantly different. A linear scalar analysis of a model equation could not detect the effect of such a term.

The Newton system analysis as described below can provide a very useful tool to help in the development and choice of numerical algorithms for the Euler and Navier-Stokes equations. Jespersen and Pulliam [5] examined the effect of using approximate Jacobians in the implicit operator for flux split schemes, see Section 1.4, and showed limited stability bounds for approximate forms. Anderson [7] employed the analysis to examine various approximate factorizations for stability. Barth [8, 9] examined various approximate Jacobians for Riemann solvers and MUSCL differencing schemes and in [10] he employed the analysis to examine new splittings of the Euler fluxes to produce a class of efficient implicit solvers.

The system analysis as outlined in [5] retains the system matrix form of the equations while applying Fourier analysis. Recast Eq. (1.11) as

$$\begin{aligned} M(Q^n)\left(Q^{n+1} - Q^n\right) : &= [I + h\delta_x A^n][I + h\delta_y B^n]\left(Q^{n+1} - Q^n\right) \\ &= -h(\delta_x E^n + \delta_y F^n) := P(Q^n) \end{aligned}$$

Rearranging terms

$$Q^{n+1} = Q^n + M(Q^n)^{-1}P(Q^n) = G(Q^n)$$

we can examine the stability and convergence by looking at $G'(Q^n)$. At steady-state $G(Q^*) = 0$ and $G'(Q^*) = I + M(Q^*)^{-1}P'(Q^*)$ so that we have

$$\begin{aligned}\lambda(G) : [M(Q^*) + P'(Q^*)]\,\overline{v} &= \lambda M(Q^*)\overline{v} \\ L\overline{v} &= \lambda K\overline{v}\end{aligned}$$

The analysis now includes the full system with $L = M + P$ and $K = M$. For the unfactored scheme $L = M + P = I$, for the factored scheme $L = M + P = I +$ cross term error and in other cases $M + P$ may involve approximate flux Jacobians, different spatial operators on the implicit M and explicit P sides and other approximations to the implicit operator. In the best case $L = M + P = I$ and as $h \to \infty$ we have full Newton.

Considering constant in space and time A, B, and C, let $\overline{v} = e^{i(j\theta + k\phi + l\alpha)}\overline{w}$, for Fourier analysis and replace derivatives by their Fourier signature, e.g. $\delta_x\overline{v} = i(\sin\theta\Delta x/\Delta x)\,\overline{w} = \lambda_x\overline{w}$.

Combining terms we have

$$\widehat{L}\overline{w} = \lambda\widehat{K}\overline{w}$$

where $\widehat{L}$ and $\widehat{K}$ are functions of θ, ϕ and α. A generalized eigenvalue problem is then solved over a spectrum of wave numbers θ, ϕ, α. It should be stressed that this analysis is by no means the nost general possible. One has to choose a finite spectrum of wave numbers, the assumption of constant in space and time A, B, and C and periodicity is another limiting factor. The analysis, though, does provide a next step past linear scalar analysis since the effect of the coupled system enters in through the eigensystem (eigenvalues and eigenvectors) of A, B, and C.

1.3.5 Artificial Dissipation

In solving practical problems with central difference schemes artificial dissipation [11] must be added to ensure stability and enhance robustness. There are many forms of artificial dissipation and these is an equivalence between upwind schemes and forms of artificial dissipation, see [11]. We consider artificial dissipation here in reference to the way it modifies the $\lambda's$ associated with the spatial differencing. Artificial dissipation applied to both the explicit and implicit operators can have the form

$$\begin{aligned}[I + h\delta_x A^n - hD_x^i][I + h\delta_y B^n - hD_y^i]\left(Q^{n+1} - Q^n\right) &= \\ -h[\delta_x E^n + \delta_y F^n] - h[D_x^e + D_y^e]\end{aligned}$$

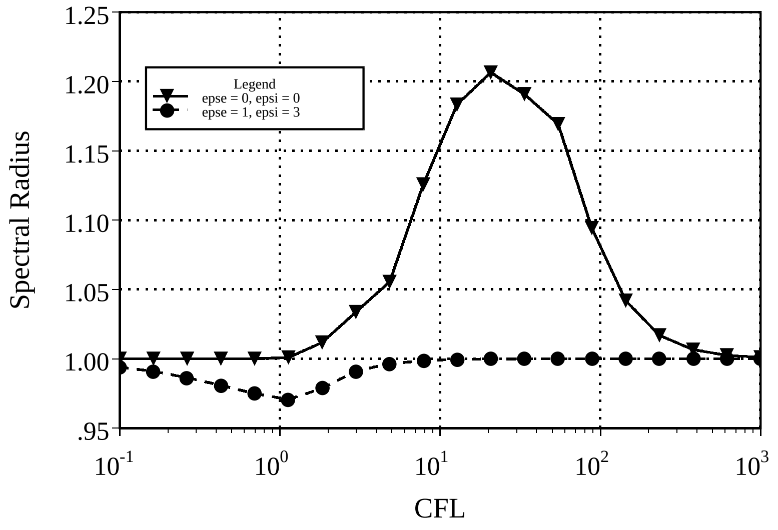

Figure 1.1: Newton System Analysis for 3D Factored Scheme

where for our purposes here we choose

$$D_x^e = -\epsilon_e(\nabla_x\Delta_x)^2 \quad D_x^i = -\epsilon_i\nabla_x\Delta_x$$

with, for example,

$$\nabla_x\, u_{j,k} = \frac{u_{j,k} - u_{j-1,k}}{\Delta x} \quad \text{and} \quad \Delta_x\, u_{j,k} = \frac{u_{j+1,k} - u_{j,k}}{\Delta x} \tag{1.13}$$

The effect of the artificial dissipation is to add a real part to the Fourier signature of the differencing eigenvalues. The coefficients ϵ_e and ϵ_i are usually chosen to be order 1. An example of Newton system analysis for the 3D factored 1^{st} order Euler implicit schemes is given in Fig. 1.1. A typical result without artificial dissipation and with artificial dissipation is shown. The addition of the dissipation produces a stability region, while the case of no dissipation is unconditionally unstable, as expected. The linear analysis result of instability for the three-dimensional factored algorithm is well known and has often been pointed out as a weakness of the factored schemes. In practice, though, for nonlinear problems in the presence of some form of dissipation (whether it is added artificial or inherent numerical, such as in upwind schemes, or physical resulting from resolved viscous terms) the algorithm has never behaved any more restrictiely than its two-dimensional linearly unconditionally stable counterpart. As we shall see below, there are alternative schemes which avoid the unconditional stability, but in practical problems they behave no differently than the three-dimensional factored scheme.

1.4 Flux-Vector Splitting

The concept of splitting operators or fluxes based on the eigensystem of the Euler equations leads naturally to the flux vector or flux difference split schemes which are currently the most popular in terms of research and application to shock capturing methods. One of the earliest schemes is due to Steger and Warming [12] where they constructed new fluxes F^+, F^- which were upwind differenced in a stable fashion based on the sign of the eigenvalues of the Jacobians $\tilde{A}^+, \tilde{A}^-$.

The approach taken is to split the eigenvalue matrix Λ of the flux Jacobians into two matrices, one with all positive elements and the other with all negative elements. Then the similarity transformations T_x or T_y are used to form new matrices A^+, A^- and B^+, B^-. Formally,

$$A = T_x \Lambda_x T_x^{-1} = T_x(\Lambda_x^+ + \Lambda_x^-)T_x^{-1} = A^+ + A^-$$

with

$$\Lambda_x^{\pm} = \frac{\Lambda_x \pm |\Lambda_x|}{2}$$

Here, $|\Lambda|$ implies that we take the absolute values of the elements of Λ. The two matrices, A^+ and A^- have by construction all nonnegative and all nonpositive eigenvalues, respectively.

New flux vectors can be constructed as

$$\begin{aligned} E =& \quad AQ = (A^+ + A^-)Q = E^+ + E^- \\ F =& \quad BQ = (B^+ + B^-)Q = F^+ + F^- \end{aligned}$$

A general form of the flux vector can be written as

$$F = \frac{\rho}{2\gamma}\begin{bmatrix} 2(\gamma-1)\lambda_1 + \lambda_3 + \lambda_4 \\ 2(\gamma-1)\lambda_1 u + \lambda_3(u+c) + \lambda_4(u-c) \\ 2(\gamma-1)\lambda_1 v + \lambda_3 v + \lambda_4 v \\ (\gamma-1)\lambda_1(u^2+v^2) + (\lambda_3/2)[(u+c)^2+v^2] + (\lambda_4/2)[(u-c)^2+v^2] + w \end{bmatrix}$$

where $w = \frac{(3-\gamma)(\lambda_3+\lambda_4)c^2}{2(\gamma-1)}$ with $\lambda_1 = u, \lambda_3 = u + c$, and $\lambda_4 = u - c$ recovering the flux vector E of Eq. (1.2).

The flux vectors F^+, F^- are formed by inserting $\lambda_i = \lambda_i^+$ and $\lambda_i = \lambda_i^-$, respectively, where for example $\lambda_i^{\pm} = (\lambda_i \pm |\lambda_i|)/2$.

The Steger-Warming flux splitting suffers from discontinuous derivatives of the fluxes at zeros of the eigenvalues (i.e. stagnation or sonic points) and one can smooth out the discontinuities by the modification $\lambda_i^{\pm} = (\lambda_i \pm (\lambda_i^2 + \epsilon^2)^{\frac{1}{2}}/2$ for small ϵ. An alternate flux splitting is proposed by Van Leer [13] where $F^{\pm}$ are given in terms of a local one-dimensional Mach number $M = \frac{u}{c}$ where

$$\begin{aligned} F^+ &= F, \quad F^- = 0 \quad \text{for } M \leq 1 \\ F^+ &= 0, \quad F^- = F \quad \text{for } M \leq -1 \end{aligned}$$

and for $|M| < 1$

$$F^{\pm} = \begin{bmatrix} f^{\pm} \\ f^{\pm}[(\gamma-1)u \pm 2c]/\gamma \\ f^{\pm}v \\ f^{\pm}[((\gamma-1)u \pm 2c)^2/2(\gamma^2-1) + v^2/2] \end{bmatrix}$$

with $f^{\pm} = \pm\rho c\left[(M \pm 1)/2\right]^2$. This flux is continuously differentiable at sonic and stagnation points.

The splitting of the fluxes into terms with Jacobians with either all positive or all negative characteristic speeds allows us to choose upwind differences for each operator. For the positive terms a backward difference can be used and for the negative terms a forward difference is applied. Different type of spatial differencing can now be used for each of the new flux vectors. The one-sided difference operators are usually either first order accurate, Eq. (1.13), or second order accurate

$$\delta_x^b\, u_{j,k} = \frac{\frac{3}{2}\,u_{j,k} - 2\,u_{j-1,k} + \frac{1}{2}\,u_{j-2,k}}{\Delta x} \qquad \delta_x^f u_{j,k} = \frac{-\frac{3}{2}\,u_{j,k} + 2\,u_{j+1,k} - \frac{1}{2}\,u_{j+2,k}}{\Delta x}$$

An unfactored 1st order Euler implicit algorithm can be written as

$$\begin{aligned}\left[I + h\delta_x^b\tilde{A}^+ + \delta_x^f\tilde{A}^- + \delta_y^b\tilde{B}^+ + \delta_y^f\tilde{B}^- + \delta_z^b\tilde{C}^+ + \delta_z^f\tilde{C}^-\right]\left(Q^{n+1} - Q^n\right) &= \\ -h\left(\delta_x^b E^+ + \delta_x^f E^- + \delta_y^b F^+ + \delta_y^f F^- + \delta_z^b G^+ + \delta_z^f G^-\right) &= R^n\end{aligned}$$

where, e.g., $\tilde{A}^{\pm}$ is the flux Jacobian of $E^{\pm}$. Note that $A^{\pm} \neq \tilde{A}^{\pm}$ and in fact if $A^{\pm}$ were used, instability could could result, see [12] and [5]. The full unfactored algorithm can be reduced by factoring as we did before. One possibility is a three-factor scheme

$$\left[I + h\delta_x^b\tilde{A}^+ + \delta_x^f\tilde{A}^-\right]\left[I + \delta_y^b\tilde{B}^+ + \delta_y^f\tilde{B}^-\right]\left[I + \delta_z^b\tilde{C}^+ + \delta_z^f\tilde{C}^-\right]\left(Q^{n+1} - Q^n\right) = R^n$$

which requires three block tridiagonal inversions, and is equivalent to central differencing and added artificial dissipation, [11]. An alternative is a two factor scheme where all the positive terms and all the negative terms are lumped together,

$$\left[I + h\delta_x^b\tilde{A}^+ + \delta_y^b\tilde{B}^+ + \delta_z^b\tilde{C}^+\right]\left[I + \delta_x^f\tilde{A}^- + \delta_y^f\tilde{B}^- + \delta_z^f\tilde{C}^-\right]\left(Q^{n+1} - Q^n\right) = R^n$$

The two factor scheme produces a purely upper and lower triangular matrix system and can be solved as lower/upper sweeps. The disadvantage of the two factor scheme is that it is harder to vectorize, the recursive nature of the sweeps making the identification of a vectorizable direction difficult. In contrast, for the three factor scheme, each operator is one-dimensional and can be vectorized over one of the other directions, see [7] for more discussion.

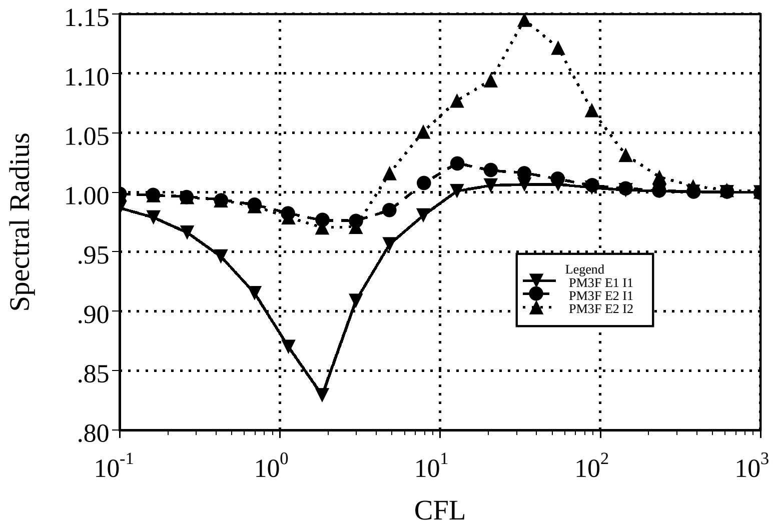

Figure 1.2: Newton System Analysis for 3 Factor ± Scheme

Newton system analysis of these schemes is presented in Fig. 1.2 and Fig. 1.3. In Figure 1.2, the three-factor scheme shows conditional stability at low CFL. In these schemes, one can choose either purely first order differences, which results in an implicit block tridiagonal system. Second order differences require an implicit block pentadiagonal solver or ane can reduce the computational work of the implicit operator but maintain second order steady-state accuracy by using first order differences in the implicit operator and second order for the explicit operator. This mixing of the order of differencing also affects the stability characteristics as shown in Fig. 1.2. (The label E1 I1 refers to 1^{st} order explicit and 1^{st} order explicit differences, etc.) Figure 1.3 shows results for the two factor scheme, where all forms are unconditionally stable with the purely 1^{st} order scheme showing the best characteristics.

1.5 F3D ± Flux Split Scheme

The three-factor implicit central difference scheme suffers from a bad reputation resulting from the linear instability as shown above. In general, for practical problems this doesn't seem to be a real restriction. Nevertheless, one would like to employ schemes which are at least stable in the linear sense. In that regard, Ying [14], developed the Factored Three-Dimensional algorithm (F3D) which employs flux splitting in one coordinate direction and central differences in the other 2 directions. This produces a two factor implicit scheme which has central difference characteristics in two coordinate directions (usually the near normal and some other cross flow direc-

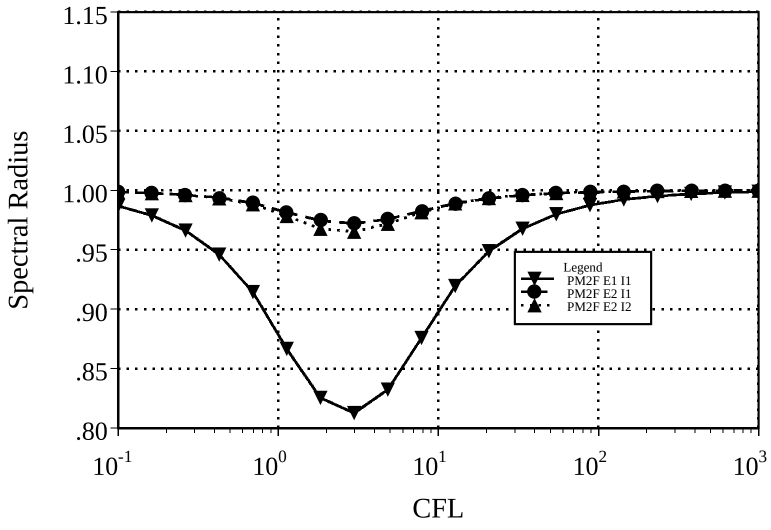

Figure 1.3: Newton System Analysis for 2 Factor $\pm$ Scheme

tion) and an upwind nature in one direction (usually chosen in the major flow direction). The advantages of this scheme is the two factor operator which can be shown to be unconditionally stable in the linear constant coefficient case and the upwind nature of one of the operators which is usually chosen in the direction perpendicular to a shock or flow disconinuity. Consider, for example, a blunt cone at angle of attack. The F3D scheme would use flux splitting in the axial direction, and central differences in the circumferential and body normal directions. Note that in inviscid supersonic axial flow, the scheme could reduce to pure supersonic marching, which can be very efficient.

Ying [14] choose the x direction to flux split and the resulting equations can be written as

$$Q_t + \delta_x^b E^+ + \delta_x^f E^- + \delta_y F + \delta_z G$$

where first or second order differences can be employed for δ_x^b and δ_x^f and second order central differences for δ_y and δ_z. The "delta" form of Euler implicit time differencing is given as

$$\begin{aligned}\left[I + h\delta_x^b \tilde{A}^+ + \delta_y B\right] \quad \left[I + \delta_x^f \tilde{A}^- + \delta_z C\right] (Q^{n+1} - Q^n) = \\ -h \quad \left(\delta_x^b E^+ + \delta_x^f E^- + \delta_y F + \delta_z G\right) = 0\end{aligned}$$

where the $\delta_x^b \tilde{A}^+$ implicit operator is placed with the y operator and the $\delta_x^f \tilde{A}^-$ operator is placed with the z operator. This produces a two-factor scheme which is lower block diagonal in x coupled with block tridiagonal in y for the first operator, which can be solved by sweeping in x within an

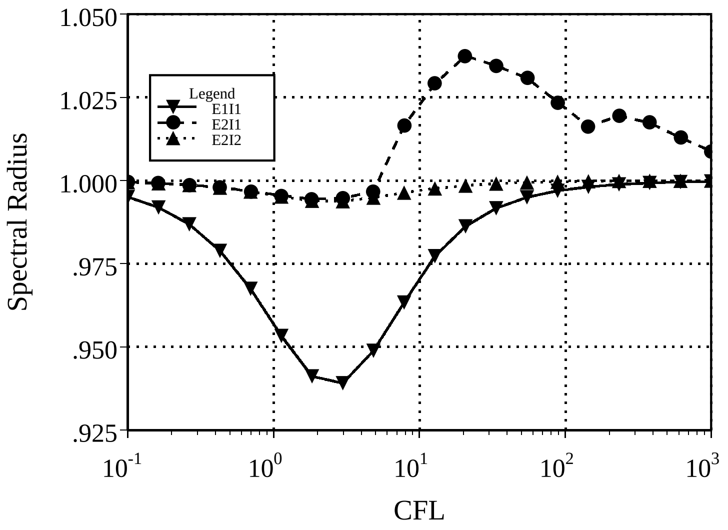

Figure 1.4: Newton System Analysis for F3D scheme

LU decomposition in y. The z operator is handled similarly except that it is upper block diagonal in x and block tridiagonal in z. This produces an efficient algorithm which does not suffer directly from the three-factor linear instability. The first operator can be vectorized in z while the second is vectorized in y.

Figure 1.4 shows Newton system analysis for the F3D scheme. Results are shown for 1^{st} order upwind in x and all 2^{nd} order differencing. To enhance the efficiency, i.e. reduce the band width of the implicit x component of the algorithm, 1^{st} order upwind differences can be used on the implicit side and 2^{nd} order on the implicit side, resulting in a 2^{nd} order steady-state. Unfortunately, this produces limited stability, although the stability range may still be in the useful region. The above results are shown for no added artificial dissipation in the central y and z directions. Adding artificial dissipation improves the stability in much the same way as for the fully central three-factor algorithm.

1.6 Summary

A class of implicit approximate factorization schemes has been examined for stability and convergence characteristics. In general, all the schemes suffer from some limited stability or asymptotic convergence restriction. Practical schemes will almost always fall within this class. The unconditional instability of the three-dimensional factored scheme is one end of the spectrum, where conditional stability can be achieved with added artificial dissipation. The F3D scheme avoids the unconditional instability, but in the end has similar convergence characteristics. Full Newton schemes are currently being pursued (see [15] and [16]) for the Euler and Navier-Stokes equations with some success. These efforts are more restricted by the computer resources than by any numerical analysis considerations, such as stability or consistency. At present, though, schemes such as F3D and the 3D factored method are the most useful and practical.

Bibliography

[1] Jameson, A., Schmidt, W. and Turkel, E., (1981), *Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes*, AIAA 81-1259, AIAA 14th Fluid and Plasma Dynamics Conference, Palo Alto.

[2] Jespersen, D. C., (1984), *Recent Developments in Multigrid Methods for the Steady Euler Equations*, Lecture Notes for Lecture Series on Computational Fluid Dynamics, von Kármán Institute, Rhode-St-Genese, Belgium.

[3] Martinelli, L. and Jameson, A., (1988), *Validation of a Multigrid Method for the Reynolds Averaged Equations*, AIAA-88-0414, AIAA 26th Aerospace Sciences Meeting, Reno, NV.

[4] Jespersen, D. C., (1985), *A Time-Accurate Multi-Grid Algorithm*, AIAA 85-1493-CP, Proceedings of the AIAA 7th Computational Fluid Dynamics Conference, Cincinnati, OH.

[5] Jespersen, D. C. and Pulliam, T. H., (1983), *Approximate Newton Methods and Flux Vector Splitting*, qyAIAA Paper No. 83-1899.

[6] Pulliam, T. H., (1985), *Efficient Solution Methods for The Navier-Stokes Equations*, Lecture Notes for the von Kármán Institute For Fluid Dynamics Lecture Series : Numerical Techniques for Viscous Flow Computation In Turbomachinery Bladings, von Kármán Institute, Rhode-St-Genese, Belgium.

[7] Anderson, W. K., (1986), *Implicit Multigrid Algorithms for the Three-Dimensional Flux Split Euler Equations*, Ph.D. Thesis, Mississippi State University.

[8] Barth, T. J., (1987), *Analysis of Implicit Local Linearization Techniques for Upwind and TVD Algorithms*, AIAA-87-0595, AIAA 25th Aerospace Sciences Meeting, Reno, NV.

[9] Barth, T. J., (1988), *Implicit Linearization Procedures for Upwind Algorithms*, Proceedings of the International Conference on Computational Engineering Science, Atlanta, GA, Vol. 2.

[10] Barth, T. J. and Steger, J. L., (1985), *A Fast Efficient Implicit Scheme for the Gasdynamics Equations Using A Matrix Reduction Technique*, AIAA-85-0439, AIAA 23rd Aerospace Sciences Meeting, Reno, NV.

[11] Pulliam, T. H., (1985), *Artificial Dissipation Models for the Euler Equations*, AIAA J. 24, No 12.

[12] Steger, J. L. and Warming, R. F., (1981), *Flux Vector Splitting of the Inviscid Gas Dynamic Equations with Applications to Finite Difference Methods*, J. Comp. Phys. 40, pp. 263-293.

[13] van Leer, B., (1983), *Flux-Vector Splitting for the Euler Equations*, Eighth International Conference on Numerical Methods in Fluid Dynamics, Springer Lecture Notes in Physics no. 170, ed. E. Krause.

[14] Ying, S. X., (1986), *Three-Dimensional Approximately Factored Schemes for Equations in Gasdynamics*, Ph.D. Thesis, Stanford University.

[15] Beam, R. M. and Bailey, H. E., (1988), *Newton's Method for the Navier-Stokes Equations*, Proceedings of the International Conference on Computational Engineering Science, Atlanta, GA, Vol. 2.

[16] Venkatakrishnan, V., (1988), *Newton Solution of Inviscid and Viscous Problems*, AIAA-88-0413, AIAA 26th Aerospace Sciences Meeting, Reno, NV.